

# Affordable High Powered Clustered Computing for Aerospace Simulation

W. McMillan, M. Woodgate, B.E. Richards, B.J. Gribben, K.J. Badcock, C.A. Masson and F. Cantariti  
Aerospace Engineering Report 9911

Aerospace Engineering Department, University of Glasgow, Glasgow G12 8QQ, U.K.

Engineering  
PERIODICALS

U5000

## Abstract

Motivated by a lack of sufficient local and national computing facilities for computational fluid dynamics simulations, the Affordable Systems Computing Unit (ASCU) was established to investigate low cost alternatives. The options considered have all involved cluster computing, a term which refers to the grouping of a number of components into a managed system capable of running both serial and parallel applications. Past work by the Unit has demonstrated the significant improvement in the efficiency of a Network of Workstations when management software is employed to scavenge spare cycles and schedule tasks, and has also investigated the use of a managed network for parallel CFD. The present work aims to extend this effort to a higher performance cluster based on commodity processors used for dedicated batch processing. The performance of the cluster has proved to be extremely cost effective, producing a 3 Gigaflops plus peak performance for less than 25K U.K. pounds sterling at current market prices. The experience gained on this system in terms of single node performance, message passing and parallel performance will be discussed. In particular, comparisons with the performance of other systems will be made. A large scale CFD simulation achieved using the new cluster will be presented to demonstrate the potential of commodity processor based parallel computers for aerodynamic simulation.

## 1 Introduction

The computational requirements for increased fidelity of modelling activities in areas such as Computational Fluid Dynamics, Rotorcraft Dynamics and Low Speed Aerodynamics provides a requirement for High Performance Computing (HPC). This was identified in the early 1990's as a key requirement for continued successful development and research into these areas. Consequently, an urgent requirement for a local HPC environment to support aerospace research was identified.

Computationally, the requirements can be divided into the serial and parallel, and batch and interactive tasks. To accommodate these requirements, a managed cluster of Silicon Graphics Unix workstations was implemented.

These workstations incorporated features such as LSF<sup>1</sup> load balancing software with PVM[1] message passing. This facility provided a good platform for serial, parallel, interactive and batch activities. However, as the complexity of the problems being considered increases, so the high performance computing requirements for the solution of these problems also increase. In response a project has been instigated to implement a high performance cluster dedicated to large parallel calculations, which does not include a requirement for an interactive capability. To achieve this a 16-node parallel machine, following the Beowulf model<sup>2</sup> was selected.

The concept of clustered workstations seeks to produce high performance computing systems from off-the-shelf components, the advantage being a resultant cost appreciably lower than traditional parallel computers. The improving capabilities of PC's have meant that they offer similar performance to workstations but at a significantly lower cost. Therefore, large improvements in the price-performance of parallel systems may be derived through the use of PC subsystems. The Pile-of-PC's (pronounced 'pop-see') approach is being explored by NASA CEDIS, Los Alamos National Laboratory, Caltech and JPL through the Beowulf Parallel Workstation<sup>2, 3</sup> program. The objective of the program is to provide order-of-magnitude increases in disk capacity, memory and bandwidth at lower costs compared to many traditional parallel computers such as the Cray T3D and IBM SP2 for specific problems. As reported in HPCWire<sup>3</sup> a sixteen node Pentium Pro Beowulf cluster costing US \$50K, running the Linux operating system and a FastEthernet interconnect achieved a sustained performance of 1.26 Gigaflops with a theoretical peak of 3.2 Gigaflops. Furthermore, an incidental feature of this cluster is the larger total memory (2 Gbytes) and disk storage (80 Gbytes) capabilities which provide cost effective solutions when compared with commercially available workstations. Such systems have also been demonstrated to be scaleable with two 16-node 'machines' being combined at SuperComputing '96<sup>4</sup>. A further and more 'extreme' example of this scalability was demonstrated by the 4536 node (2

<sup>1</sup><http://www.platform.com>

<sup>2</sup><http://cesdis1.gsfc.nasa.gov/beowulf/>

<sup>3</sup><http://www.cacr.caltech.edu/research/beowulf/HPCwire/>

<sup>4</sup><http://www.supercomp.org/sc96/>

Pentium Pro processors per node) US Department of Energy Accelerated Strategic Computing Initiative machine (ASCI Red)<sup>5</sup> which at a cost of \$55M achieved 1.3 Teraflops in 1997.

The type of facility that was proposed by the Beowulf project at Glasgow, took the form of a single user machine for parallel computing. However, this model can also provide a multi-user environment supporting serial and parallel applications providing supercomputer performance with the addition of load management software. Consequently, this type of machine was adopted as the model for the Department of Aerospace's facility. In addition, recent developments in Ethernet communications technology in the form of Fast and Gigabit-Ethernet offers an attractive alternative to ATM for high speed networking.

The current paper describes the utility of this system for large scale aerodynamic simulations. The key issues to be considered are processor and message passing performance. Results are presented to illustrate these features. Following this, several examples are given to demonstrate the system capabilities for aerodynamic simulations. First, the system components, which are all standard commodity items available in street shops, are described.

## 2 Description of Pentium cluster system

The components are all *off-the-shelf* items purchased from a local hardware vendor. The system consists of a sixteen port 100BaseTx Fast Ethernet switch, a standard Pentium Pro 200 PC (which forms the front end machine) and sixteen Pentium Pro 200 PC cases (with no monitors, keyboards or mice). Each of the machines is fitted with a 100BaseTx network card. The components of the machine were named after Jupiter (the front-end machine) and its sixteen moons (all other machines). They are assembled as shown in Figure 1. The basic specification of the machines is

- PCI Local Bus Intel Triton 440FX Chipset Pentium Pro Motherboard. (This is a dual processor motherboard).
- one or two Pentium Pro Processors operating at 200Mhz
- 256Kb Secondary Cache RAM (on chip)
- 256Mb Ram
- 3.6Gb EIDE hard disk drive
- 3com 3C905 10/100 UTP Ethernet Cards
- ATI 4Mb SGRAM 3D Expression Graphics Card

<sup>5</sup><http://www.sandia.gov/ASCI/Red>

All of the communications with the outside world go through the front end machine, which has two processors and an additional 3com 3C905 10/100 UTP Ethernet Card. The two cards of the front end are connected to the outside world and the file server respectively. The file server is used for serving all software and user home areas and features two processors and four 6.4 Gb EIDE hard disk drives. This also has two Ethernet cards, one of which is connected to the front end and one directly to the Fast Ethernet switch. Of the remaining 15 machines, four have two processors each and eleven have single processors. Each of these is connected to the Fast Ethernet switch.

The front end is typically used for the preparation of batch jobs. The remaining sixteen nodes are used exclusively for the execution of batch jobs. The total memory available on the system for batch jobs is therefore in excess of 4 Gb and the theoretical peak performance is 4.0 Gflops.

Keeping to the spirit of building a system at low cost, most of the software used is public domain. The operating system used is the public domain UNIX clone Linux<sup>6</sup>. The PGCC<sup>7</sup> and GNU<sup>8</sup> suite of compilers are available, together with Message Passing Interface (MPI)[2] and Parallel Virtual Machine (PVM)[1] for the control of parallel jobs. The only propriety software used is the load management software, Load Sharing Facility (LSF)<sup>9</sup> which acts as an intelligent queuing system, and the NaG-Ware<sup>10</sup> Fortran 90 compilers.

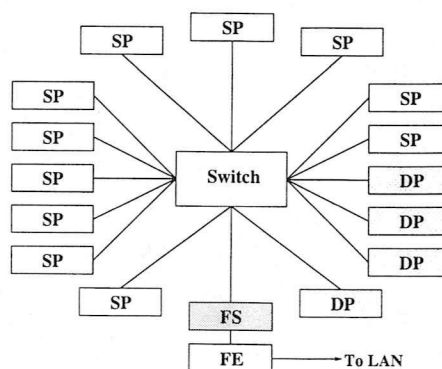


Figure 1: Configuration of Pentium system: SP - Single Processor Node DP - Dual Processor Node FE - Front End machine FS - File Server

Benchmark serial and parallel CFD simulations are used to test the system. The flow solver used is the University of Glasgow's multiblock solver PMB[4]. This

<sup>6</sup><http://www.redhat.com/>

<sup>7</sup><http://www.goof.com/pcg/>

<sup>8</sup><http://www.gnu.ai.mit.edu/>

<sup>9</sup><http://www.platform.com>

<sup>10</sup><http://www.nag.com/>

Given name	Processor	Speed (MHz)	CPU time (s)
PII	Pentium PII 300 oc 450	450	284
Octane	Silicon Graphics R10000	195	331
DEC	Digital 21164	433	346
Sun	Sun U10	300	433
Jupiter	Intel Pentium Pro	200	551
O2	Silicon Graphics R5000sc	200	857
SGI	Silicon Graphics R5000	180	931

Table 1: *Single node performance for benchmark CFD simulation*

code uses an implicit time discretisation of a finite volume formulation of the Euler or Navier-Stokes equations. The major computational tasks are to calculate numerical fluxes and their Jacobians and to solve a sparse linear system using a preconditioned Krylov sub-space method. For the preconditioning stage the blocks are decoupled, allowing for an efficient method when executed in parallel[5].

The parallel implementation of the flow solver is achieved using a coarse-grain data parallel approach with message passing. Overlapping grid blocks are employed with two rows of 'halo' cells associated with each internal block boundary. After each time step the updated solution is copied to these halo cells from the corresponding cells in the adjacent block, such that each block has the necessary information to form the residual vectors and Jacobian matrices for the next time step. If blocks sharing a common boundary reside on different processors, then the copying of data is enabled using message passing. The message passing library PVM is used.

### 3 Serial performance

A serial benchmark is shown to put the single-node performance of Jupiter in the context of other available systems. The simulation was a three-dimensional Euler calculation around a delta wing, the grid consisting of 34,153 cells. The CPU time required for the calculation on a number of different processors is recorded in Table 1, and plotted in Figure 2. These results provide a comparison of the benchmark which has been optimised for the Pentium processor, however, experiments with problems of different sizes and various compiler options do indicate a similarity in relative performances. The performance of the Jupiter nodes compares well with UNIX workstations, and is likely to offer the best price-performance given current market prices. For example, at current market prices a considerable increase in performance is possible with the use of an Intel Pentium II processor overclocked to 450 MHz for less than 1K U.K. pounds sterling.

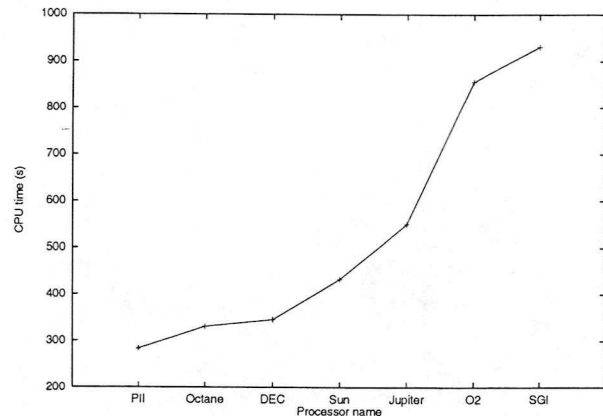


Figure 2: *Single node performance for benchmark CFD simulation*

### 4 Message passing performance

In this section, the performance of the inter-processor communication method employed during parallel calculations is investigated.

The most commonly used alternative to PVM is MPI, which is also in the public domain. In order to provide a comparison between these two message passing options a series of tests was undertaken in which the relative performance was investigated for a range of message sizes. These tests were performed on the Jupiter cluster using GNU<sup>11</sup> supplied test software. Figure 3 shows the results obtained in which the transfer rate of the data is plotted as a function of the message size in bytes. From the figure it is evident that PVM and MPI have similar performance, in terms of the transfer rates obtained, over the range of message size considered. However, due to the fixed packet size forwarded by PVM and MPI, there is a variation in the throughput when the message is the same order as the packet size. Note that a 64Kbyte socket buffer size on the Ethernet cards has been used for the MPI test.

The performance of the communication network itself is clearly of importance to the efficiency of parallel applications. The FastEthernet network employed in the Jupiter cluster is expected to be significantly faster than the standard Ethernet which is commonly employed with clusters of workstations. To verify this, tests were performed on the Jupiter cluster and using Ethernet connected SGI R5000 workstations. The transfer rate was recorded for varying communication (or packet) size. The throughput versus message size results obtained are shown in Figure 4. As can be seen from the figure the maximum throughput for the Jupiter connections is approximately 11.2MBps and for the SG 1.1MBps clearly illustrating the advantage of using FastEthernet instead of Ethernet technology. (Mucci and London [3] found the throughput on the Cray T3E to be approximately

<sup>11</sup><http://www.gnu.ai.mit.edu/>

200MBps.)

Finally, whilst slower than a dedicated supercomputer, the FastEthernet technology was found to be sufficient for the use of the fully-implicit multi-block code used at Glasgow. This is so, since the associated communication times required during a given calculation accounts for no more than 5-6% of the total computational time required for the calculations presented in Section 5 below. This relatively small (and acceptable) decrease in system performance is associated with the communication requirements for the fully-implicit multi-block code which may be broadly split into two areas. The first of these areas relates to the generic *gather scatter* (an all-to-all communication, where every process has data for every other) associated with the global inner product calculations which are totally latency based. The second operation is related to the transfer of halo data which is dependent on the block interface size, which in-turn, is proportional to the throughput of the communication medium.

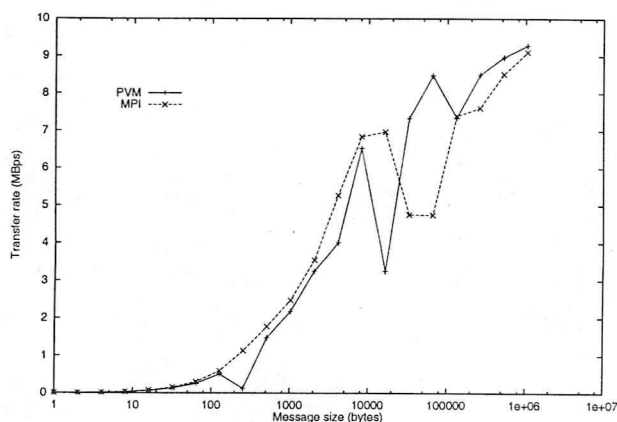


Figure 3: Comparison of PVM and MPI message passing utilities on the Jupiter cluster.

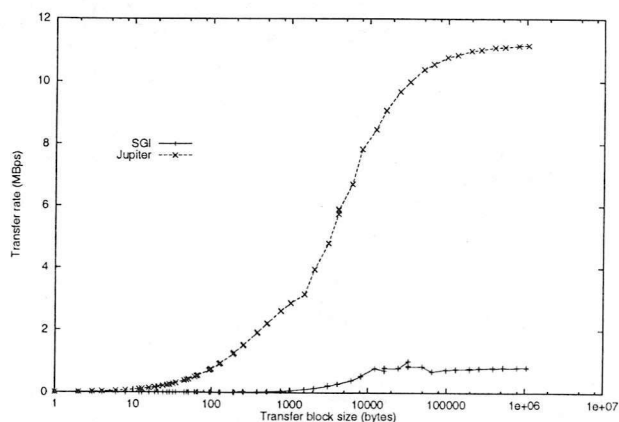


Figure 4: Comparison of the Ethernet throughput graphs for the SGI and Jupiter clusters.

Grid	Cells	Nodes	Jupiter	SP2	T3E
Coarse	24,576	1	1.0 (8.4)	0.82	–
Medium	196,608	2	1.0 (80)	1.16	–
Medium	196,608	16	1.0	–	0.51
Fine	1,572,864	16	1.0 (690)	–	–

Table 2: Relative Calculation Time compared with Jupiter (execution time in node minutes for M6 wing calculation on Jupiter is shown in brackets).

## 5 Large Scale Applications

In order to evaluate the performance of the Jupiter cluster against the alternatives of 'traditional' parallel computers, a series of calculations were performed on Jupiter and two other machines, namely an IBM-SP2 located at Daresbury Laboratory and a Cray T3E located at Edinburgh Parallel Computing Centre. The calculation of the transonic flow around the ONERA M6 wing using PMB was the chosen application. Three different grid levels were employed, at each level a reduction of six orders of magnitude in the L2 norm of the residual was chosen as the convergence criterion. The number of grid cells at each level is included in Table 2. Each of the machines used has possessed 256Mbytes of main memory on each node. One, two and sixteen nodes were used for the coarse, medium and fine calculations respectively. These are the minimum number of nodes possible due to memory considerations.

Problems were encountered running the 16 node calculations on the SP2. This was thought to be a problem associated with the way the current code spawns tasks. No major effort was made to correct this. Problems were also encountered running the fine grid on 16 nodes of the T3E. This was thought to be due to the larger executable size produced on this machine. The relative performance of the machines is indicated in Table 2. It is clear that the single node performance of the Jupiter and SP2 are comparable. The calculation on 16 nodes for the medium grid is twice as quick as Jupiter, reflecting the faster processors. However, no major influence is observed from the faster communications on the SP2 and T3E, reflecting the high parallel efficiencies achieved on the Jupiter cluster. In comparison, the Jupiter cluster gives excellent performance considering its low cost.

The calculated pressure contours on the upper surface of the wing are shown in Figure 5. Comparison between calculated pressure coefficient values on the wing surface and experimental data is shown in Figure 6. Good agreement is shown.

A complex geometry application is presented to further demonstrate the potential of the Pentium cluster for the examination of large-scale problems. The case considered is an F5 wing with a tip launcher and missile. Figure 7 shows calculated surface pressure contours that took 135 minutes to calculate using 169K cells on 4 pro-



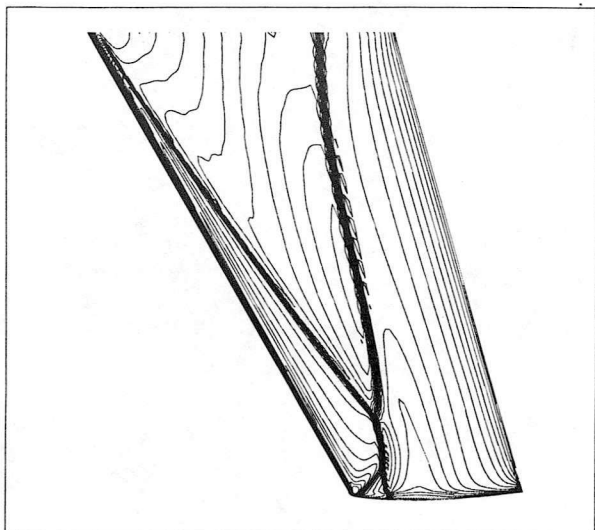


Figure 5: Calculated pressure contours on upper surface of M6 wing

cessors.

As a second example the unsteady flow around the pitching LANN[6] transport type wing was calculated. The free stream Mach number of the case is 0.82, the mean incidence  $0.6^\circ$ , the amplitude  $0.25^\circ$  and the frequency of the sinusoidal pitching 0.204.

Two grids were used for the calculations, the fine one with 597849 points and the coarser with 78125 points. The fine grid calculation was run on 8 processors and the coarser grid on one processor. Twenty time steps per cycle was used in both cases. The mean pressure contours are shown for the fine grid upper surface in Figure 8. The two main features in the flow is the lambda structure shockwave. The comparison of the mean sectional pressure distributions with experiment is also shown in Figure 9, showing good agreement within the limitations of the inviscid modelling. The wall clock times for these calculations are 4 hours per cycle for the fine grid and 3 hours per cycle for the coarse grid. The increased computing time on the fine grid is exactly in proportion to the increased number of iterations required to solve for the solution at each new time step (20.9 pseudo iterations/time step compared with 14.5 on the coarse grid). This increased work is likely to be due to the improved resolution of the shock motions on the fine grid. This also again indicates that the parallel performance of the simulation is high.

## 6 Conclusions

A parallel computer, based on 16 commodity PC processors, has been employed for CFD calculations. The outstanding price performance promised by this technology has been demonstrated. The speed of the individual processors was tested against workstations from a num-

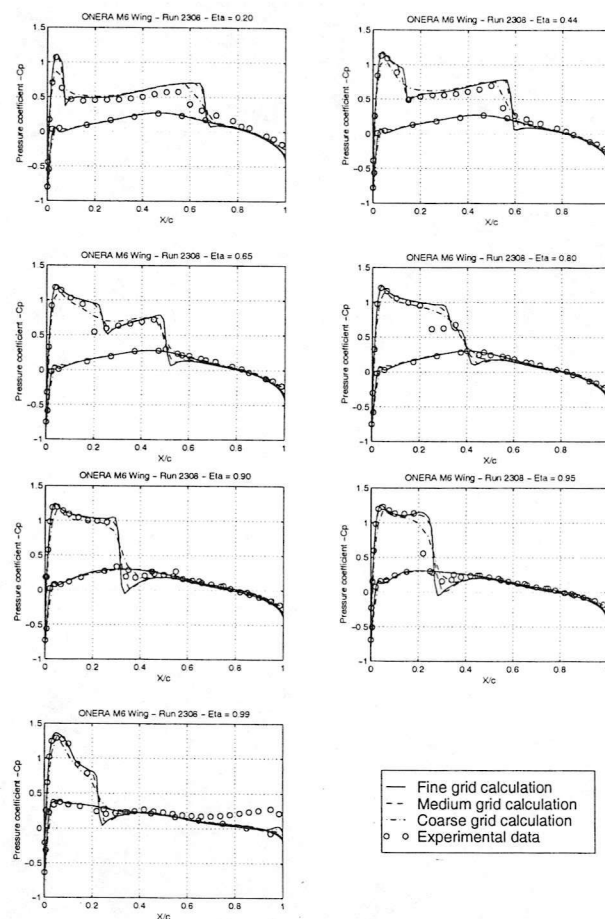


Figure 6: Surface pressure coefficient, M6 wing

ber of vendors. The performance of the PC processors is in the same range as that offered by the more expensive workstations. Operating under a sophisticated proprietary batch scheduling software suite, Jupiter is a powerful resource for high throughput serial tasks. This flexibility is useful, but the main purpose of the machine is as a parallel computer. The Fast Ethernet technology employed for inter-processor communications was an order of magnitude faster than standard Ethernet, indicating the potential of the system for parallel applications. Timing tests were performed using a large-scale parallel CFD test case which required all 16 processors, comparison being made with times using other parallel computers. The times using Jupiter were again in the same range as those on the more expensive machines for the benchmark code considered here. As a further example, the results from a test case involving a complex geometry are also presented. The capability of a Beowulf-class parallel computer for parallel CFD simulation has been clearly demonstrated.

The ability of commodity items to achieve significant computing power at a cost which is an order of magnitude lower than conventional supercomputers, significantly increases the potential of simulation for aerodynamic studies.

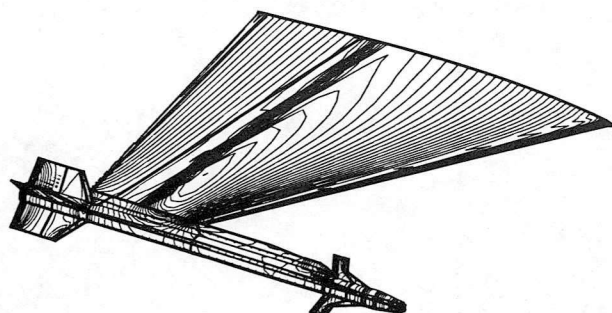


Figure 7: Surface pressure contours for wing-launcher-missile calculation.

## Acknowledgements

The authors would like to thank Michael Henshaw (BAe) for making available the grid for the NLR F5 wing with launcher and missile. This work has been partially supported by BAe contract (SP050104458), Defence and Evaluation Research Agency (DERA) contract FRNI C/107, EPSRC grant GR/K55455 and Scottish Enterprise.

## References

- [1] V. Sunderam, G. Giest, J. Dongarra and R. Manchek, 'PVM: A Framework for Parallel Distributed Computing', *Journal of Concurrency: Practice and Experience*, **2**, 315–339, (1990).
- [2] M.P.I. Forum, 'MPI: A Message-Passing Interface Standard', *Journal of Supercomputer Applications*, **8**, (1994).
- [3] P.J. Mucci, K. London, 'The MPBench Report', *University of Tennessee, Knoxville, Tech. Report ut-cs-98-394*, (1998).
- [4] L. Dubuc, F. Cantariti, M. Woodgate, B. Gribben, K.J. Badcock and B.E. Richards, 'Solution of the Unsteady Euler Equations Using an Implicit Dual-Time Method', *AIAA Journal*, **36**, 1417–1424, (1998).
- [5] K.J. Badcock, W. McMillan, M.A. Woodgate, B.J. Gribben, S. Porter and B.E. Richards, 'Integration of an Implicit Multiblock Code into a Workstation

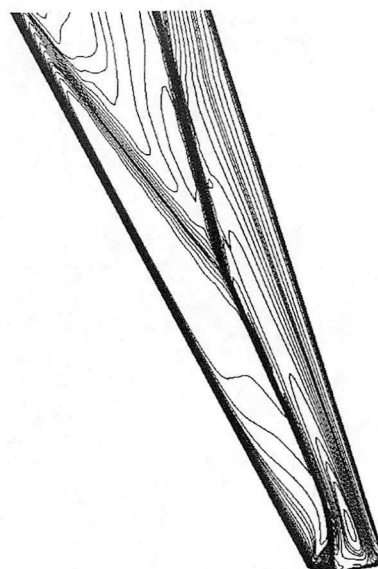


Figure 8: Mean pressure contours for the LANN wing calculation.

Cluster Environment', in *Parallel Computational Fluid Dynamics: Algorithms and Results using Advanced Computers*, ed., P. Schiano et al., pp. 408–415. Elsevier Science B.V. Amsterdam, (1996).

- [6] R.J. Zwaan, 'LANN wing, pitching oscillation', in *Compendium of Unsteady Aerodynamic Measurements - Addendum I, AGARD Technical Report 702*, (1985)

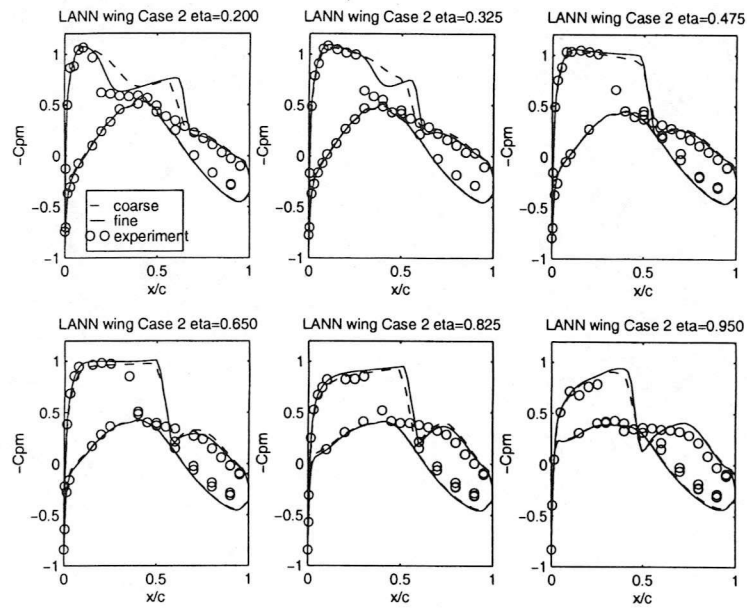


Figure 9: Comparison of the Mean Sectional Pressure distributions for the LANN wing.

